The first of a five-part series of papers on IMS/VS, this paper discusses the architecture, goals, and objectives of that information management system, the purpose of which is to facilitate Data Base/Data Communication applications. Subsequent papers present data base facilities, batch processing, data communication, and transaction processing in greater depth.

The information management system IMS/VS Part I: General structure and operation

by W. C. McGee

Computer applications in which users access a data base through remote terminals are known generally as Data Base/Data Communication (DB/DC) applications. The emergence of DB/DC applications in recent years has stimulated the development of a new kind of software, the *generalized data base management system.*[1] Such a system is intended to simplify the installation and operation of DB/DC applications. In the past, these systems took two forms: the *self-contained* system and the *host-language* system.[2,3] The self-contained system provides its services through a high-level nonprocedural language that is usually unique to the system. The host-language system does the same through common procedural languages, such as COBOL and PL/I. Newer systems tend to provide both kinds of interfaces, so that the terms "self-contained" and "host-language" now more accurately refer to facilities within a system than to systems per se.

The information management system IMS/VS[4] is a generalized data base management system that is considered to be primarily a host-language system. Application programs are written by a user in assembler language and other procedural languages, and they call on the system as required for data and communication services. Programs are incorporated into the system and are subsequently invoked in response to various user activities. Thus the response characteristics of the system are determined primarily by the user's programs. A generalized query language is also provided in IMS/VS, giving the system some self-contained capability.

IMS/VS had its origins in the mid-1960s in a batch-only system known as Data Language/I (DL/I).[1] This system provided for accessing hierarchical data structures from COBOL programs running on a System/360 computer. An on-line component, Information Control System/Data Language/I (ICS/DL/I), was added later to permit terminal access to data and to permit multiple programs to operate concurrently.

DL/I evolved into IMS/360,[5] an IBM program product for the System/360 released in 1969. IMS/360 provided for the definition, creation, and manipulation of hierarchically structured data bases with two implementations: hierarchic sequential (HSAM) and hierarchic indexed sequential (HISAM). Supported terminals included the IBM 1050 and the IBM 2740, with terminal communications controlled by the Basic Telecommunications Access Method (BTAM). The system provided the capability to initiate user-written application programs in response to incoming messages, as well as a batch initiation capability. Programs called for system services through the DL/I language. The system ran as one or more regions or partitions of Operating System/360, with one region controlling the operation of the system and the remaining regions executing application programs.

IMS/360 has evolved into IMS/360 Version 2,[6] and thence into IMS/VS. The functional capabilities of the system have been significantly enhanced, although the basic system architecture has remained the same. The performance of the system has also been enhanced through the redesign of some system components, and through the addition of facilities that permit the system to be more closely adapted to particular environments.

This paper presents a status report on the current capabilities of IMS/VS, which runs on System/370 under OS/VS1—discussed in Reference 7—and OS/VS2—discussed in Reference 8. To illustrate the discussion, reference is made to the IMS/VS installation at the Automobile Club of Michigan in Dearborn, Michigan.

The paper is divided into five major parts. Part I presents the general structure and operation of IMS/VS, and Part II discusses data base facilities. Batch processing facilities are presented in Part III, data communication is discussed in Part IV, and transaction processing is considered in Part V.

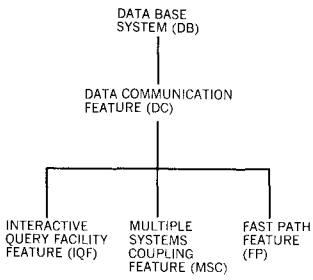
Structure and operation

IMS/VS is a program product for System/370 that runs under the OS/VS1 and OS/VS2 operating systems. IMS/VS is offered as a basic system with optional features. The basic system, the Data Base (DB) system, provides facilities for defining, creating, and

maintaining IMS/VS data bases, and for running user-written programs in the batch mode to process the data bases. The features include the following:

- Data Communication (DC) provides for the transmission of messages between IMS/VS and remote terminals, and for the invocation of user-written programs to process incoming messages against IMS/VS data bases.
- Interactive Query Facility (IQF)[9] provides for the generalized processing of terminal-entered queries against IMS/VS data bases.
- Multiple Systems Coupling (MSC) provides for the routing of messages between two or more IMS/VS systems that run in the same or different CPUs.
- Fast Path (FP)[10] provides message processing and data base facilities for high-volume, limited-function DB/DC applications.

Figure 1 IMS/VS structure



The IQF, MSC, and FP features require the DC feature, which in turn requires the DB system as shown in Figure 1. The DB system with the DC feature is referred to as the Data Base/Data Communication (DB/DC) system.

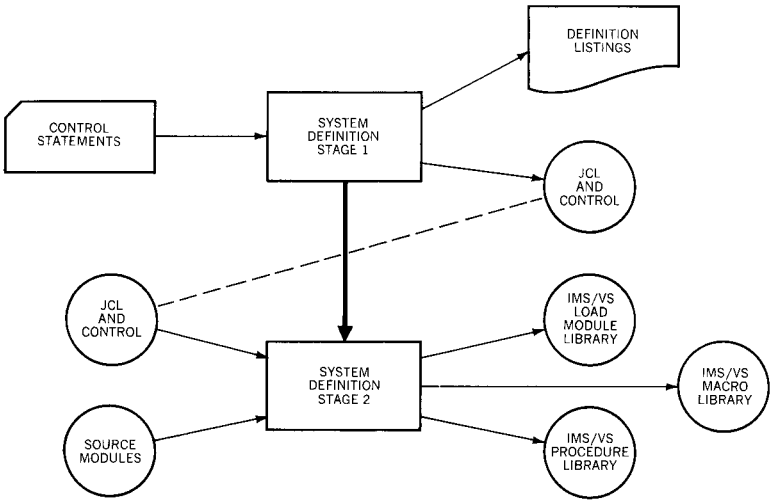
The program product is shipped as a set of assembler language modules and a system definition procedure. Before the product can be used, the procedure must be run to convert the supplied modules into a running system that is tailored to each user's specific environment and applications. The procedure accepts user-prepared control statements that provide the following information:

- Hardware and software environment in which the defined system is to run.
- Names and other attributes for user-supplied objects, such as application programs and message types.
- Space required in main storage and direct access storage for system buffers and control blocks.
- Operating options, such as the location of control blocks during system operation.

The procedure illustrated in Figure 2 produces a tailored set of load modules and places them in the IMS/VS load module library. In addition, copies of macro definitions and job control language procedures needed for system operation are placed in the IMS/VS macro library and procedure library.

The system definition procedure may be repeated to define multiple IMS/VS systems. The modules generated by a given run of the procedure have a user-specified suffix appended to their

Figure 2 System definition



names so that multiple systems may coexist in the load module library. The particular system to be used in a given execution is selected by the system operator.

IMS/VS provides two types of system execution: batch and on-line. A DB system is capable of batch executions only, whereas a DB/DC system permits both batch and on-line executions.

A *batch execution* of IMS/VS is a normal operating system job. The job is initiated by the operator through a job request that specifies the IMS/VS *region controller* module as the program to be executed, and allocates to the job the data sets for the IMS/VS data bases to be accessed. The job request also specifies, as a parameter to be passed to the region controller, the name of a user-written program to be run in this system execution.

When an operating system region becomes available, the job is scheduled for execution. The region controller is fetched from the load module library and is given control. The region controller performs the following operations:

1. Loads from IMS/VS control block libraries the control blocks that define the user's program and the data bases to be accessed.
2. Loads from the IMS/VS load module library the action modules required in this execution of the system.
3. Loads the user's program from the IMS/VS application program library.
4. Transfers control to the user's program.

While the user's program is running, it may issue calls to the resident action modules to retrieve data from and update data in IMS/VS data bases that are allocated to a job. When the program completes, it returns control to the region controller, which in turn relinquishes control to the operating system, thereby ending the job. The region in which a batch execution runs is called a *batch region*, to distinguish it from the region types used in an on-line execution.

IMS/VS provides a number of utility programs for use in data base definition, application program definition, and system operation. These utility programs are executed in a batch execution of the system (i.e., in a batch region) in exactly the same manner as user-written programs, except that they are fetched from the load module library instead of the application program library.

An *on-line execution* of IMS/VS uses one or more operating system regions. One of these regions, the *control region*, holds the IMS/VS *control program*. The remaining regions are called *dependent regions*, and are used for the user's application programs. By using multiple regions, IMS/VS takes advantage of operating system inter-region protection facilities to prevent interference among application programs and the control program.

The control region is started by the system console command START IMS. This command creates an operating system region and starts the IMS/VS control program as a system task. The command also allocates to the region the data sets for the on-line data bases and message queues, and the communication facilities to be used in receiving and sending messages to terminals. (When VTAM is used as the telecommunications access method, communication facilities are allocated to a separate VTAM region, and IMS/VS communication requests are made through the VTAM region.)

Once the control program is started, IMS/VS commands may be entered from the IMS/VS master terminal to cause the enabling, disabling, and display of various resources within the on-line execution. A master terminal command is also used to terminate the on-line execution.

A master terminal command is used to enable input from user terminals. Inputs can be of four types:

- *Commands* are requests for the control program to perform system-related functions. These commands are a subset of the commands that can be entered from the master terminal.
- *Switched messages* are messages that are to be routed to another terminal. Switched messages are queued by destina-

tion, and are transmitted as the destination terminal is available.

- *Transactions* are messages to be processed by users' application programs. Transactions carry a transaction type code, and are queued by transaction type as they are received. With the Fast Path feature, a transaction type may be designated as a fast path transaction type, and transactions of that type are queued by associated application program.
- *Queries* are messages that request the generalized retrieval of data from on-line data bases and its display at the originating terminal. Queries are queued in the same manner as transactions.

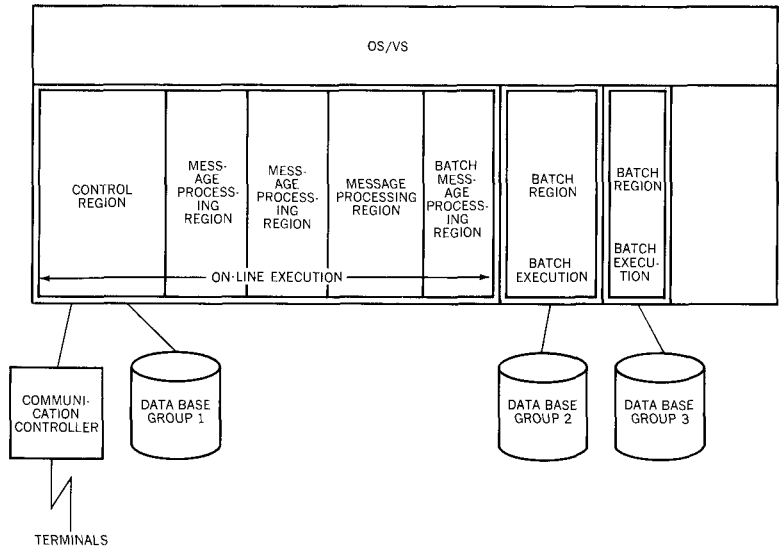
To process transactions and queries, one or more dependent regions must be started. These are of two types: *message processing regions* and *batch message processing regions.* Message processing regions are used to execute application programs in response to incoming messages and queries, and batch message processing regions are used for operator-scheduled programs that require access to the on-line data bases or the message queues. Both types of region operate as normal operating system jobs.

A message processing region job request specifies the IMS/VS region controller as the program to be executed. When the region controller obtains control, it initializes the region and waits for work from the control region. When the message processing region is no longer required, it may be terminated by a master terminal command that causes the region controller to terminate, thus ending the job in that region.

Whenever a message processing region is available, the control program selects a waiting transaction and schedules the appropriate user-written *message processing program* into the region to process it. The message processing program typically fetches the transaction from the queue, accesses the on-line data bases, and sends a response message back to the terminal that originated the transaction. The program can also send messages to other terminals and to other transaction queues. When the program has processed one transaction, it may ask for additional transactions of the same type and process them in like manner. When the queue is empty, the program returns control to the region controller, which then indicates to the control program that it is available for rescheduling.

If the control program selects a query to be processed, the Interactive Query Facility (IQF) program is scheduled and executed in a manner similar to a message processing program.

Figure 3 Example system executions



A batch message processing region job request also specifies the IMS/VS region controller as the program to be executed. In addition, it specifies the name of a user-written *batch message processing program* to be invoked and, optionally, the name of a transaction queue to be accessed. When the region controller gets control, it invokes the batch message processing program. When the message processing program terminates, it returns control to the region controller, which in turn ends the job. Except for the manner in which it is scheduled, a batch message processing program has all the capabilities of a message processing program.

For processing fast path transactions, one or more *message-driven regions* must be started. Such a region is analogous to the message processing region, except that the job request that starts it specifies a user-written *message-driven program* to be given control when the job starts. This program retains control of the region for the duration of the on-line execution or until stopped by master terminal command.

A message-driven program obtains transactions from its associated transaction queue and processes them in a manner similar to a message processing program. Data services are limited to the accessing of special fast path data base types, the *main storage data base* and *data entry data base*, and communication services are limited to communication with the terminal that originated the transaction.

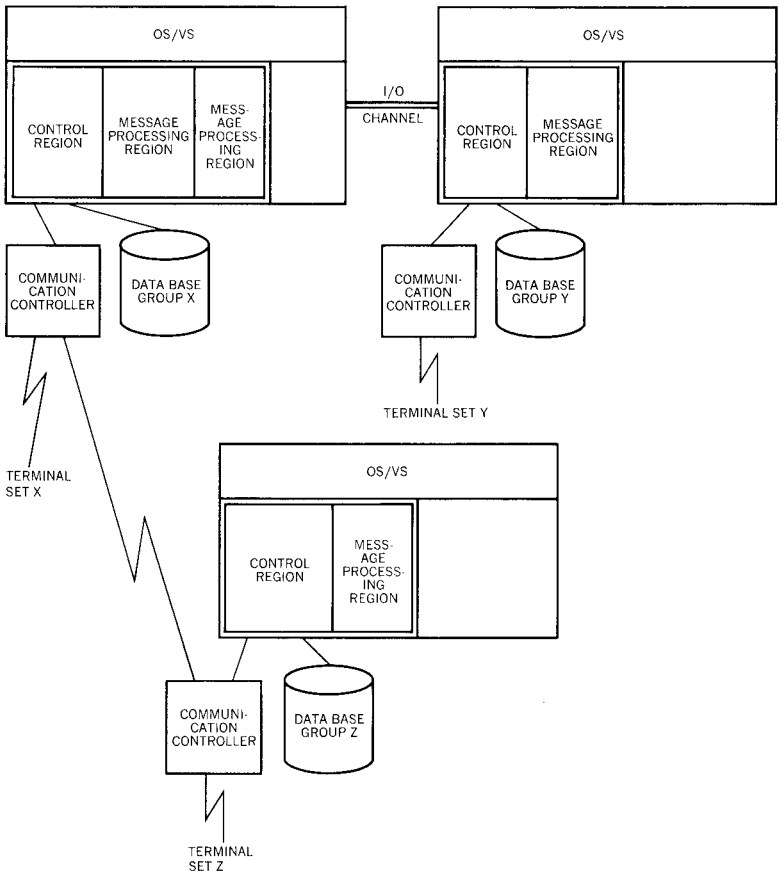
The Fast Path feature also provides for *non-message-driven regions* in an on-line execution, analogous to batch message processing regions. Non-message-driven programs can access fast path data bases that are assigned to the execution, but not the fast path transaction queues.

Executions of IMS/VS may operate with a *system log*. The log is optional for a batch execution, but it is required for an on-line execution. The log is a tape data set that is allocated to the batch region or control region, in which are recorded significant events that occur during system operation. This includes message receipt and transmission, application program scheduling, data base changes, and system checkpoints. The log is used in a variety of operational procedures, including normal and emergency startup, data base and message queue recovery, and the preparation of system usage reports.

Through the multiprogramming facilities of the operating system, one or more batch executions and one or more on-line executions may operate concurrently on the same CPU. Data bases may be allocated exclusively to one execution at a time, or to multiple executions if they perform only retrieval. Figure 3 illustrates a typical mix of IMS/VS executions.

Communication between on-line executions is provided by the Multiple Systems Coupling (MSC) feature. Executions that reside in the same CPU communicate via main storage or I/O channels. In different CPUs they communicate via I/O channels or Binary Synchronous Communication (BSC) lines. Communication consists of the sending and receiving of messages of the same type that are exchanged by terminals and the system in a single on-line execution. Input messages from terminals that are controlled by one on-line execution may be queued in the terminal and transaction queues of that execution, or they may be forwarded to another on-line execution for queuing there. Messages that are generated by application programs may similarly be queued in the same or different on-line execution. The routing of messages between on-line executions is determined by configuration and message destination definitions that are included in the system definition or by a user's exit coding. The ability of one-line executions to communicate permits the transaction processing load presented by a given terminal configuration to be shared by two or more CPUs, thus increasing the processing capacity of the configuration over that achievable with a single CPU. The same communication capability permits data bases to be dispersed geographically so that data can be placed close to primary users but still be accessible to occasional users. Figure 4 illustrates a configuration that is made possible by the Multiple Systems Coupling feature.

Figure 4 System executions with multiple system coupling



The Automobile Club of Michigan

The Automobile Club of Michigan[11] is an affiliate of the American Automobile Association, and provides service to the residents of Michigan. The Club has 1.2 million members, making it the third-largest affiliate of AAA. Services provided by the Club include emergency road service, travel services, loans, and life, auto, home, and boat insurance. The Club has issued 1.1 million auto policies to its members, the largest number of any AAA affiliate. Headquarters of the Automobile Club is in Dearborn, and there are fifty-seven branch offices located throughout the state.

The headquarters houses a computer installation that contains two System/370 computers, a model 158 and a model 168. The computers are connected by leased lines to approximately two hundred IBM 3277 terminals. During the day, the Model 168 is used for on-line applications that run under IMS/VS, and the

model 158 is used for program development and testing using the Time Sharing Option (TSO) of OS/VS2. During the swing shift, both machines are used for batch processing.

On-line applications currently installed include the following:

- Policy rating.
- Cash deposit control.
- Policy and member information display and data entry.
- Claims reporting.
- Terminal operator training.

Batch applications include the following:

- Cash payment.
- Billing.
- Policy renewal.
- Policy and member information maintenance.

The Automobile Club's data processing activities are supported by a Systems and Programming Department, a Technical Support Department, and a Data Processing Department. The Technical Support Department includes a data base administration function that is responsible for all aspects of DB/DC software, data base design, and the security and integrity of systems and procedures. The data base administrator chairs data base design reviews for all IMS projects. The Data Processing Department operates the computer facility, and includes a Communication Control Section that is responsible for data base backup and reorganization activity, and for supervising data base recovery and on-line system restart operations. The Automobile Club initially installed IMS in 1972, using IMS/360 Version 2.3. IMS/VS was installed in November 1974. The Club is currently using IMS/VS 1.0.1, running under OS/VS2 (MVS).

Summary and concluding remarks

The overall objective of a generalized data base management system is to simplify the development and running of DB/DC applications. IMS/VS achieves this objective primarily through three key sets of facilities: a high-level application programming interface, a generalized program scheduling facility, and a comprehensive set of utility programs.

IMS/VS simplifies application programming by providing a high-level programming interface in which logical data structures are limited to hierarchies, and are largely independent of underlying data organizations and storage devices. Application programming is further simplified by shifting much of the function nor-

mally found in application programs to declarative-language specifications prepared by installation specialists. This shift is consistent with the data base administrator concept, and materially reduces the traditionally high cost of application programming.

A generalized facility for scheduling application programs on the basis of external events (i.e., transactions) permits application programs to be written without regard for the mechanics of message transmission, message queuing and dequeuing, program loading and scheduling, and resource allocation. All of these functions have been successfully generalized within acceptable performance bounds.

IMS/VS provides extensive supporting facilities (such as restart, recovery, and reorganization) that are perhaps not as obvious as the main stream facilities, but which are a practical necessity in the development of DB/DC applications. In particular, IMS/VS restart and recovery facilities have permitted many users to make their DB/DC applications an integral part of their day-to-day operations without fear of excessive disruption to these operations in the event of system or program failure.

ACKNOWLEDGMENTS

The author is indebted to Mr. T. W. Klamo and his associates at the Automobile Club of Michigan for supplying the Automobile Club material that appears in this paper.

CITED REFERENCES

1. J. P. Fry and E. H. Sibley, "Evolution of data-base management systems", *ACM Computing Surveys* **8,** No. 1, 7–42 (March 1976).
2. CODASYL Systems Committee, *A Survey of Generalized Data Base Management Systems*, Technical Report (May 1969). May be obtained from the ACM.
3. CODASYL Systems Committee, *Feature Analysis of Generalized Data Base Management Systems*, Technical Report (May 1971). May be obtained from the ACM.
4. IBM Corporation, *IMS/VS Version 1 General Information Manual*, Document GH20-1260, IBM Corporation, Data Processing Division, White Plains, New York 10604.
5. IBM Corporation, *IMS/360 Application Description Manual*, Document GH20-0524 IBM Corporation, Data Processing Division, White Plains, New York 10604.
6. IBM Corporation, *IMS/360 Version 2 General Information Manual*, Document GH20-0765, IBM Corporation, Data Processing Division, White Plains, New York 10604.
7. T. F. Wheeler, Jr., "OS/VS1 concepts and philosophies," *IBM Systems Journal* **13,** No. 3, 213–229 (1974).
8. A. L. Scherr, "Functional structure of IBM virtual storage operating systems; Part II: OS/VS2-2 concepts and philosophies," *IBM Systems Journal* **12,** No. 4, 382–400 (1973).
9. IBM Corporation, *Interactive Query Facility (IQF) for IMS/360 Version 2 General Information Manual*, Document GH20-1074, IBM Corporation, Data Processing Division, White Plains, New York 10604.

10. IBM Corporation, *IMS/VS Version 1 Fast Path Feature General Information Manual*, Document GH20-9069, IBM Corporation, Data Processing Division, White Plains, New York 10604.
11. IBM Corporation, Data Processing Division, "Into high gear," *Data Processor* **18,** No. 4, 6–7 (September 1975). This article describes the Automobile Club of Michigan application. IBM Corporation, Data Processing Division, White Plains, New York 10604.

Reprint Order No. G321-5046.